REMARKS

This Amendment is responsive to the Official Action mailed on August 26, 2004. The Office Action rejected all of the pending claims, with claims 9 and 10 rejected under 35 U.S.C. 102(b) as anticipated by DeVita *et al.* (U.S. patent number 4,325,118), claims 11 and 12 rejected under 35 U.S.C. 102(b) as anticipated by Kojima *et al.* (U.S. patent number 5,880,981), claims 25-27 rejected under 35 U.S.C. 102(e) as anticipated by Pickett *et al.* (U.S. patent number 6,101,595), and claims 28-30 rejected under 35 U.S.C. 103(a) as unpatentable over Baji *et al.* (U.S. patent number 5,535,417) in view of Keichi (Japanese patent publication number 6211292). As described in the following, it is respectfully submitted that these rejections are not well founded and should be withdrawn. As also described below, several claims have been amended and a number of new claims have been added.

Claims 9 and 10

The Office Action rejected claims 9 and 10 rejected under 35 U.S.C. 102(b) as anticipated by DeVita *et al.*, specifically referring to column 1, lines 40-44, for the second and third elements of claim 9. This states "Multiple instructions require the appropriate number of bytes to be *fetched in sequence* form the program memory", where the emphasis is added. Thus, as is also explicitly noted in the Office Action (last two line of page 2), a multiple byte instruction is *not* fetched in a single fetch operation, but instead only a byte of a is provided in a given fetch, so that a multiple byte instruction *requires a corresponding number of sequential fetches*. For instance, a 3-byte instruction requires 3 sequential fetches. As described in the Background section of the present application, this is precisely the sort of problem that, in one aspect, the invention as presented in claim 9 avoids.

In contrast, as described in the Summary (beginning with its first sentence) and elsewhere in the present application, a principle aspect of the present invention allows all bytes of a maximal length instruction to be accessed simultaneously. For example, page 4, line 6-10, of the application states (emphasis added):

> In an exemplary processor embodiment, the invention is based upon performing a single 24-bit memory fetch, which *allows the user to access all three bytes of the maximum length instruction simultaneously*. This access needs only to occur once per machine cycle, which provides a relaxation of the memory access time by at least a factor of two.

That is, not only is this aspect of the present invention *not* anticipated by DeVita, but, rather, the cited portion of DeVita *teaches the opposite procedure.*

This distinction is distinction is reflected in the final element of claim 9, which states:

> a memory interface for supplying the instructions from the memory to the central processing unit, wherein all bytes of each of said instructions are supplied simultaneously in a single fetch operation.

Consequently, as shown by the added emphasis, this contrary to the sort of byte-by-byte sequential fetch cited by the Office Action.

(Although believed allowable in its original form for the reasons just stated, claim 9 has been amended to make it explicit that, in a "single fetch operation", "all bytes of each of said instructions are supplied simultaneously". Also, it should be noted that the Office Action is being inconsistent with itself: in its rejection of claim 9 it is states "a single instruction operation is performed by fetching three bytes sequentially", which is contrary with its rejection of claims 28 and 29 where it states (page 5, lines 10-11) "fetching one instruction, by definition, is done in a single fetch operation.")

Concerning the second element of claim 9, this states that "a memory for storing the instructions, wherein the instructions are stored contiguously", where this sort of contiguous storage is described, for example, at page 9, lines 9-13, or page 10, lines 16-26, of the present application. For this restriction, the Office Action again refers to column 1, lines 40-44, of DeVita. This cited portion reads, in its entirety, as:

> Multiple instructions require the appropriate number of bytes to be fetched in sequence form the program memory. Substantial time is consumed in sequentially fetching the bytes forming a single instruction, limiting instruction execution rate.

Neither in this passage nor elsewhere in DeVita can any reference to the contiguous storage of instructions be found.

Therefore, for any on the preceding reasons, it is respectfully submitted that a rejection of claim 9, along with its dependent claims, under 35 U.S.C. 102(b) as anticipated by DeVita is not well-founded and should be withdrawn.

Concerning claim 10, this is already believed allowable as it depends from claim 9. It is further believed allowable as DeVita further lacks specific disclosure for supplying all bytes of 2-byte instructions simultaneously and provides no specific disclosure of storing elements of an instruction set with instructions of various lengths (1-, 2-, and 3-bytes) contiguously in an instruction memory.

Claims 11 and 12

The Office Action rejected claims 11 and 12 rejected under 35 U.S.C. 102(b) as anticipated by Kojima *et al.* Claims 11 and 12 again recite a "a memory for storing the instructions, wherein the instructions are stored contiguously". This sort of contiguous storage is discussed above with respect to claims 9 and 10. The Office Action cites column 4, lines 26-27, of Kojima. No specific disclosure of the instructions being stored contiguously can be found at this location (or elsewhere) in the reference, only the description of a 24-bit wide ROM for storing 3-byte (and only 3-byte) instructions. Consequently, a rejection of claims 11 and 12 under 35 U.S.C. 102(b) is respectfully submitted to be in error and should be withdrawn.

Claim 11 additionally recites that the memory of the claims is a one time programmable (OTP) memory, which are discussed at page 1, line 21 *et seq.*, page 4, line 30 *et seq.*, and elsewhere in the present application. The Office Action again cites column 4, lines 26-27, of Kojima. No specific disclosure of the use of one time programmable (OTP) memories can be found at this location (or elsewhere) in the reference. Consequently, a rejection of claim 11 under 35 U.S.C. 102(b) is respectfully submitted to be in error and claim 11 is believed further allowable for this reason.

Claims 25-27

The Office Action rejected claims 25-27 rejected under 35 U.S.C. 102(e) as anticipated by Pickett *et al.* This rejection is also respectfully submitted to be in error as Pickett neither discloses nor suggests elements found in these claims.

Claims 26-30 have been amended to rationalize their reference to their base claim, claim 25, which is a method claim. Claim 25 itself has been amended to make it more explicit that each of the M columns of the memory's organization are a byte wide.

The second element of claim 25 reads "programming the instruction set into the memory, wherein the instructions are stored contiguously in the memory". The Office Action identifies the memory with element 112 ("Instruction Storage") in Figure 3 of Pickett and cites column 2, lines 50-51 and 58-59, for this element of the claim. This is believed to be incorrect for a number of reasons.

First, as described in Pickett, element 112 is a cache memory in which cache lines, formed of *individual* instructions *selected* from the *instruction set* are stored. Pickett provides no disclosure of "programming the *instruction set* into the memory"[emphasis added], only

the writing of individual elements of the set (which have been formed up into cache lines) into element 112.

(Although the various aspect of the present invention are applicable to cache memories, this neither taught nor suggested in Pickett---in particular, as described here, the use of Pickett's element 112 differs in a number of important respects from the invention presented in claim 25.)

Further, these instructions are *not* "stored contiguously in the memory". This lack of contiguous storage is shown explicitly in Pickett's Figure 5, which shows the organization of an exemplary cache line. Figure 5 shows a number of instructions (IN0, IN1, ...) taken from the instruction set. It also shows a number of predicted branch indicators (PB0, PB1). As described beginning at column 16, line 57, these predicted branch indicators *are not part of the instruction set*, but, rather, are generated by the system based upon the preceding instruction in the cache line. As shown in Figure 5, when the cache line is formed up, these are interposed between instructions from the instruction set; for example, PB0 is placed between IN1 and IN2. Thus, there is a gap between the instructions IN1 and IN2 that does not contain an element of the instruction set. Consequently, for the cited element 112, Pickett explicitly *teaches away* from "programming the instruction set into the memory, wherein the instructions are stored contiguously in the memory".

(Additionally, although Pickett uses the term "contiguous", it should be noted that the use of "contiguous" in Pickett has a very particular meaning. This is described at column 7, lines 8-11: "As used herein, the term 'group of contiguous instruction bytes' is used to refer to the instruction bytes which are provided by the instruction cache is a particular clock cycle in response to a fetch address." This is not the same as *storing* an instruction set contiguously, meaning in an adjacent manner without gaps, *within* a memory. One is related to the temporal proximity of instruction bytes as supplied, while the other relates to placement of the instruction set within the physical memory.)

Additionally, the first element on claim 25 recites "logically organizing the memory as a plurality rows of M byte-wide columns, wherein M is an integer greater than one and wherein N and M are relatively prime". The Office Action cites element 112 ("Instruction Storage) in Figure 3 of Pickett. The Pickett figure does show the element 112 divided vertically into fourths; however, as is described there beginning at column 14, line 12, what this indicates is the degree of associatively of 112 ("FIG. 3 depicts a four-way set associative cache ..."). There is no specific disclosure that the memory is organized as "M byte-wide

columns, wherein M is an integer greater than one and wherein N and M are relatively prime".

Therefore, for any on the preceding reasons, it is respectfully submitted that a rejection of claim 25, along with its dependent claims, under 35 U.S.C. 102(e) as anticipated by Pickett is not well founded and should be withdrawn.

Claims 25-27

The Office Action rejected claims 28-30 rejected under 35 U.S.C. 103(a) as unpatentable over Baji *et al.* in view of Keichi. These rejections are believed to be in error for several reasons. All of these claims are dependent claims, where claim 29 has claim 25 as its base claim and claim 28 depends from claim 26.

Concerning both of claims 28 and 29, for the step of "programming the instruction set into the memory, wherein the instructions are stored contiguously in the memory", the Office Action only cites "figure 1, 1400". This is element is referred to at column 5, line 7, as an "instruction memory"; however, Baji neither teaches nor suggests "programming the instruction set into the memory, wherein the instructions are *stored contiguously*". Claims 28 and 29 are further believed allowable on this basis.

Further, it also appears that Baji does not disclose "operating the interface whereby each of the instructions can be supplied from the memory to the central processing unit in a single fetch operation"; instead, it appears that the Office Action is instead improperly assuming that this must be the case without any support from the cited reference. (It is again noted that the "definition" of a fetch given here in the Office Action contradicts the comments made in its rejection of claim 9).

Concerning claim 30, this is already believed allowable as it depends from claim 29. It is further believed allowable for several additional reasons. In paragraph 11, the Office Action states that "With respect to claim 30, Baji discloses a one time programmable memory (figure 23, 3200, ROM)." It is respectfully submitted that this further rejection is incorrect for several reasons. First, with respect to claims 28 and 29, the Office Action has previously identified the "memory" of the claims with element 1400 of Baji's Figure 1; this is a different memory than element 2300 of Baji's Figure 23 that the Office Action identifies with the "memory" of claim 30. This is inconsistent and improper. As described beginning at column 29, line 30, element 2300 is a separate memory used for a very specific purpose (namely, for booting up the DSP) and *not* the instruction memory 1400.

Furthermore, to meet the limitations of claim 30, the "memory" of the claim needs to be a one time programmable (OTP) memory having "M byte-wide columns, wherein M is an integer greater than one". Memory is 2300 of Baji is a ROM that is *only 1-byte wide*, as is clear form both the description beginning at column 29, line 30, and from Figure 23, where the I/0 line is labeled "I/O *0-7*".

For any of these reasons, claim 30 is believed to be further allowable.

New Claims and Conclusion

A number of new claims have been added. These claims are drawn to similar aspects of the present invention as the previously pending claims, but often in differing combinations. These claims are consequently believed allowable for the reasons given above for the previously pending claims.

Reconsideration of claims 9-12 and 25-30, and consideration of new claims 31-40, is therefore requested and an early indication of their allowability is earnestly solicited.

Respectfully submitted,

MG Cl 11/24/04

Michael G. Cleveland — Date
Reg. No. 46,030

PARSONS HSUE & DE RUNTZ LLP
655 Montgomery Street, Suite 1800
San Francisco, CA 94111
(415) 318-1160 (main)
(415) 318-1163 (direct)
(415) 693-0194 (fax)